DOST: a distributed object segmentation tool

Multimed. Tools Appl. manuscript No.
(will be inserted by the editor)

DOST: A Distributed Object Segmentation Tool

Muhammad Shahid Farid · Maurizio Lucenteforte · Marco Grangetto

Received: date / Accepted: date

Abstract This paper presents a novel distributed object segmentation framework that allows one to extract potentially large coherent objects from digital images. The proposed approach requires minimum user supervision and permits to segment the objects accurately. It works in three steps starting with the user input in form of few mouse clicks on the target object. First, based on user input, the statistical characteristics of the target distributed object are modeled with Gaussian mixture model. This model serves as the primary segmentation of the object. In the second step, the segmentation result is refined by performing connected component analysis to reduce false positives. In the final step the resulting segmentation map is dilated to select the neighboring pixels that are potentially incorrectly classified; this allows us to recast the segmentation as a graph partitioning problem that can be solved using the well-known graph cut technique. Extensive experiments have been carried out on heterogeneous images to test the accuracy of the proposed method for the segmentation of various types of distributed objects. Examples of application of proposed technique in remote sensing to segment roads and rivers from aerial images are also presented. The visual and objective evaluation and comparison with the existing techniques show that the proposed tool can deliver optimal performance when applied to tough object segmentation tasks.

Keywords Object segmentation · Graph cuts · Gaussian mixture model · Connected component analysis

M.S. Farid (*The Corresponding Author*)
Punjab University College of Information Technology
University of the Punjab, Pakistan
E-mail: shahid@pucit.edu.pk

M.S. Farid, M. Lucenteforte, M. Grangetto
Dipartimento di Informatica,
Università Degli Studi di Torino, Turin, Italy

1 Introduction

Image segmentation is the process of dividing an image into multiple perceptually meaningful segments. Image segmentation is a primary research area in computer vision and image processing due to its vast practical applications in various fields of life. It has been a fundamental process in many medical imaging techniques [1, 2] and is essential in object recognition [3–7], video surveillance [8, 9], change detection [10–12], image coding [13–16], and image retrieval [17, 18]. Object segmentation (or foreground segmentation) is used to segment a particular object from a digital image. These methods, also known as *interactive image segmentation*, are semi-automatic and requires human user assistance to obtain satisfactory results however, automatic object segmentation is imperative.

Existing object segmentation techniques require different level of user interaction to create a *trimap* which divides the image into three regions, namely *definite foreground*, *definite background* and *unknown*. The segmentation is then reduced to the classification of the unknown pixels as foreground or background. Object segmentation approaches like [19, 20] require the user to draw a boundary e.g., circle, rectangle, around the target object and treat the region outside the specified boundary as background and the bounded region is processed to extract the target object. Other techniques e.g., [21–23] expect few user strokes marked on the foreground and/or background regions which are used to model and then segment the two regions. Some methods e.g., [24, 25, 41] require the user to draw edges around the object boundary which serves as a rough segmentation and is refined though matting tools [26]. These techniques, however, are not effective in segmenting distributed objects covering the entire image, e.g. fences, roads, rivers, railings or multiple coherent targets scattered in the image, e.g. flowers of the same kind. Indeed, in such cases the definition of the trimap these methods depend upon turns to be a very time consuming, tedious and error prone task. Fig. 1 shows few sample images with distributed objects: fence, flowers, river, and road; it is evident that marking the target region boundaries is impractical in these cases. The proposed 'Distributed Object Segmentation Tool' (DOST) aims at following goals: accurate segmentation of distributed objects with simplistic and minimum user interaction. Preliminary results of this research are reported in [27].

In this paper we propose a novel object segmentation technique to extract targets that can be spread on the whole image. With little user interaction DOST estimates the statistical characteristics of the target object using Gaussian mixture model and creates an initial segmentation; this is then refined using *Connected Component Analysis* (CCA) to remove false positives. The resulting segmentation is still likely to be inaccurate especially around object boundaries. Therefore, a further refinement step is carried out: using dilation the image regions around the partial segmentation results are selected as they are likely to be classified incorrectly. Finally, graph cut is applied to estimate the correct boundaries. The main advantages of the proposed DOST are:

- It can effectively segment distributed or scattered objects which are challenging tasks for the most existing object segmentation technique. Moreover, DOST can also be used to segment single coherent objects.

Fig. 1 Few images from test dataset with distributed objects: fence, flowers, river and road.

– Minimal user assistance is required to train the statistical model characterizing the target object; few mouse clicks on the target object are enough as opposed to more complex and time consuming marking requirements adopted by other techniques. Moreover, these marking techniques may not work in images where the target object is spanning over the whole image, for example, a thin fence.
– The undesired background is automatically inferred in DOST without any user intervention. That is, the user is not required to provide seeds for the background.
– Graph based segmentation is used only as the last refinements step based on a trimap that is created automatically; the unknown region in the trimap is kept as limited as possible to ensure fast segmentation.
– The proposed method is quite generic is able to work on heterogeneous images to segment various types of objects, e.g., fences, flowers, roads, rivers, railings, etc. The results show the effectiveness of the proposed technique.

The rest of the paper is organized as follows: Section 2 reviews the state of the art interactive object segmentation methods. Section 3 describes the initial object segmentation and its rectification through Connected Component Analysis (CCA) and Section 4 describes the Graph-Cut based segmentation refinement. Experimental evaluation is presented in Section 5 and the conclusions are drawn in Section 6.

2 Object Segmentation Background

Image segmentation is indeed a pixel labeling problem and various solutions to this problem have been proposed in literature mainly based on level-set, neural networks, watershed principle and graph partitioning methods [28–32]. Object segmentation is a binary labeling problem that partitions the image into two regions; foreground and background. *Image Matting* [32] is another segmentation related research area that estimates the *alpha channel* for each pixel in the image. The alpha estimate is a value $\in [0, 1]$, where 0 means definite background and 1 means definite foreground and other values between 0 and 1 represent the degree a pixel belong to the foreground. Image matting has attracted much research in the past due to its application in image and video editing in film industry where it is used to change the background of the scene in a seamless fashion. Image matting is also known as 'soft segmentation' and if the alpha values are constrained to 0 and 1 only, the problem is reduces to binary segmentation.

Intelligent scissors [33] allowed quick and more accurate object segmentation. It is an interactive segmentation tool that exploits the object contours. It asks the user

to provide few seeds on the object boundary and uses 'live-wire' boundary detection [34] to find the optimal contour between the seed points. The user can improve the segmentation by providing more seeds in an interactive way. The segmentation results of intelligent scissors are usually satisfactory with adequate user assistance. Corel Knockout 2 [35] is considered to be one of the best object segmentation tool [24] and is fairly easy to use unlike its Lasso and Magic Wand tools. Knockout needs the object trimap describing the foreground, background and the unknown regions. It allows the user to specify the trimap with the pencil tool, the user marks the outline and inside of the target object which form the three regions of the trimap. The alpha value of a pixel in unknown region is estimated by computing the ratio of the distance between the pixel color to the object color and background color [36] in RGB color space.

Graph based segmentation has gained immense popularity during the last decade due to its superior performance [37]. Initially, graph cuts were able to solve binary labeling problem - foreground-background segmentation - which was generalized for *N-dimension* by Boykov et al. [38]. In graph based segmentation approaches, the image is represented as weighted undirected graph and the segmentation is solved by finding global minimum of energy function defined over the graph. Like graph cut, Bayesian matting approach [24] models the foreground and the background from the user specified trimap using the mixture of Gaussians however, it assumes the fractional blending of the foreground and background in the boundary region. It estimates the blending value know as 'alpha value' of the complex boundary regions like hair strands to create plausible compositions. According to [19], a considerable user effort is required to obtain satisfactory results. A review of well known graph based image segmentation techniques can be found in [39].

Lazy Snapping [40] is another interactive segmentation approach that requires the user to provide input by marking few lines on the target object and on the background. The segmentation is performed through graph cut technique [38] and object boundaries are constructed in the form of polygons. To further improve the segmentation, the user can move the polygon vertices or can add few points to refine the object boundary. The Lazy Snapping shows adequate results however, user is fatigued in terms of boundary editing which is quite similar to seeding step in Intelligent scissors method.

Rother et al. [19] proposed iterative graph cuts for foreground segmentation. Their technique 'GrabCut' proposed an iterative solution to graph cut optimization based on initial hard segmentation. Further, to improve the fine segmentation of object boundaries border matting [41] is used. GrabCut requires the user to draw a rectangle around the target object and treats the rest of the region as background which is then represented with *Gaussian mixture models* (GMM). Analogously, GMM is also estimated for the target region which is then refined through global optimization. Segmentation results can be improved further with user interaction which updates the two foreground and background GMM. GrabCut allows the user to extract target object with much ease, however to refine the segmentation user may need to feedback the system. The segmentation quality of GrabCut is significantly better than its ancestors. A new update strategy has been proposed in [20], based on user input instead of performing the entire segmentation process again, it is limited to the erroneous re-

gion. The new update strategy thus avoids the change in already perfectly segmented regions. Other approaches using GrabCut or mixtures of Gaussians to model foreground and background regions can be found in [42–46].

GrowCut [22] solves the labeling problem though bi-directional, deterministic cellular automaton [47]. In the automaton the nodes called bacteria, represent the pixels and starting with the seed each bacteria attacks its neighbors to grow its region. Active Cuts (AC) [48] proposed a novel approach to solve the max-flow min-cut problem that significantly improved the efficiency of graph cut method. A GPU based solution to max-flow min-cut algorithm is proposed in [49], Parallel and distributed solutions to graph cuts are proposed in [50–53] to achieve real time segmentation. A comprehensive description of energy functions that can be minimized using graph cuts is given in [54] and an overview of graph cuts may be found in [55].

The object segmentation method proposed in [56] exploits the superpixels and Bayes classification to extract the target objects. It uses the simple linear iterative clustering (SLIC) [57] to cluster the features of the color image which are used to in the GrabCut to obtain the acute segmentation. Moreover, it applies the Gaussian mixture model (GMM) to SLIC features and the energy function used in segmentation is based on GMM based SLIC. In [58], a convolutional network based algorithm is proposed for semantic segmentation from images. The algorithm proposed in [59] obtains an initial segmentation based on user specified foreground though a modified energy function described in [19]. The local prior distribution is then used to improve the segmentation, unlike the iterative approach proposed in many similar techniques. The foreground segmentation technique proposed in [60] also uses the superpixels [57] to achieve the task. The algorithm is based on probabilistic framework which uses the low-level image features to obtain unary and pairwise probabilities. The Gaussian mixture models are used to model the foreground and background in the images and the segmentation is formulated as an inference problem over a two-dimensional Markov Random Field (MRF). The adaptive constraint propagation cut (ACP Cut) is an interactive image segmentation technique proposed in [61]. From the input provided by the user, the algorithm uses semisupervised kernel matrix learning which adaptively propagates this information into the whole image.

Most of the segmentation techniques described above are able to accurately segment a single coherent object from colored photographs and require trimap as input. Trimap initialization in case of a single object is viable, however it is a tiresome and painfully difficult job in case of distributed objects (see Fig. 1). The following section describes the proposed binary segmentation technique that is capable of accurately segmenting distributed objects in color images.

3 Distributed Object Segmentation

The proposed distributed object segmentation technique is a multi-stage algorithm that starts with a little user input in the form of few clicks on the target object. Based on this input the object is then segmented automatically. Fig. 2 shows a block level diagram of the proposed algorithm and each step is detailed in the following sections.

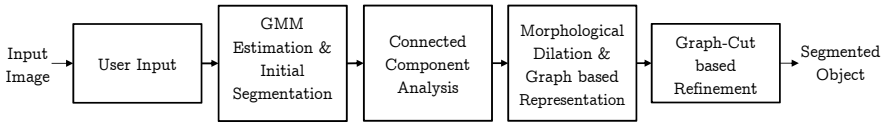


Fig. 2 Block diagram of proposed algorithm.

3.1 User Input and Target Object Modeling

The user can provide input either by marking few points with mouse click or by drawing few scribbles on the target object. Based on the selected points the color model of the object is estimated and used to classify the rest of the image pixels, leading to an initial segmentation map. Let P be set of n number of points marked by the user on input image I. Then, the number of marked pixels is automatically increased to improve the modeling accuracy. In DOST, we propose to include the κ-neighboring points of each $p_i \in P$. It turns out that a total of $\kappa^2 n$ points representative of the target objects are collected.

Gaussian Mixture Model (GMM) is a general and accurate approach to represent the statistical properties of the marked pixels. A GMM is a parametric probabilistic model given by a weighted summation of Gaussian density functions. In particular, the likelihood of a pixel x, given the Gaussian mixture model $\mathcal{G}$ is defined as:

$$p(\mathrm{x} \mid \mathcal{G}) = \sum_{i=1}^{K} \pi_i g(\mathrm{x} \mid \mu_i, \Sigma_i) \tag{1}$$

where K is the number of Gaussian components in the model and π_i, μ_i and Σ_i are the weight, mean and covariance of i-th Gaussian component with

$$g(\mathrm{x} \mid \mu_i, \Sigma_i) = \frac{1}{\sqrt{(2\pi)^3 \mid \Sigma_i \mid}} \exp(-\frac{1}{2}(\mathrm{x}-\mu_i)^\top \Sigma_i^{-1}(\mathrm{x}-\mu_i)) \tag{2}$$

The model parameters

$$\{\{\pi_1, \pi_2, \cdots \pi_K\}, \{\mu_1, \mu_2, \cdots \mu_K\}, \{\Sigma_1, \Sigma_2, \cdots \Sigma_K\}\}$$

can be estimated through expectation maximization (EM) algorithm [62, 63] with K_0 components, where K_0 is a large number greater than expected number of components which is then iteratively optimized through Minimum Description Length (MDL) method [64]. EM algorithm is an iterative procedure to efficiently compute the *maximum-likelihood* (ML) estimates of unobserved or hidden data (aka latent variables). The model parameters π, μ, Σ are randomly initialized and EM algorithm then iteratively improves the estimates. Each iteration consists of following two steps:

1. E: Expectation step computes the expected log-likelihood value based on the current model parameters $\mathcal{G} : \{\pi_k, \mu_k, \Sigma_k\}$ as follows:

$$P(z_i = k \mid \mathbf{x}_i) = \pi_k \log g(\mathbf{x}_i \mid \mu_k, \Sigma_k)$$

2. M: Maximization step computes the model parameters maximizing the log-likelihood computed in the previous step:

$$\mu_k = \frac{\sum_i P(z_i = k \mid \mathbf{x}_i)\mathbf{x}_i}{\sum_i P(z_i = k \mid \mathbf{x}_i)},$$
$$\pi_k = \frac{1}{n}\sum_i P(z_i = k \mid \mathbf{x}_i),$$
$$\Sigma_k = \frac{\sum_i P(z_i = k \mid \mathbf{x}_i)(\mathbf{x}_i - \mu_k)(\mathbf{x}_i - \mu_k)^\top}{\sum_i P(z_i = k \mid \mathbf{x}_i)}$$

where hidden variable $z \in \{1, \ldots, K\}$. A good literature on model parameter estimation methods with a detailed description of EM algorithm can be found in [65].

A general problem in all clustering approaches is the selection of the number of clusters K to use. We use *Minimum Description Length* (MDL) method [64] proposed by Rissanen to estimate the the model order. The MDL principal selects the model that results in minimum code length of the data and the model parameters. Indeed, it attempts to minimize the following function $\mathcal{C}$:

$$\mathcal{C}(\mathcal{G}, K) = -\log p(\mathrm{x} \mid \mathcal{G}) + \frac{1}{2} L \log(M) \tag{3}$$

where L is the code length of model parameters $\mathcal{G}$. M is the size of the data i.e., $M = 3\kappa^2 n$ as there are $\kappa^2 n$ points and each is represented with 3 values - R, G, B. The minimum value of $\mathcal{C}$ in Eq. 3 is achieved by minimizing the log-likelihood term (increasing the maximum-likelihood (ML)) and the code length required to describe the model parameters and the data.

Starting with K_0 components, EM algorithm is used to estimate $\mathcal{G}$ and MDL is used to find the optimal order. After each iteration, the two most similar clusters are merged decrementing K by one and EM algorithm is applied again to find the model parameters. This process is repeated until $K = 1$. Finally, $\mathcal{G}$ and K corresponding to smallest MDL value $\mathcal{C}$ are selected as optimal model parameters. The reader is referred to [66–70] for details on exploiting MDL framework for clustering.

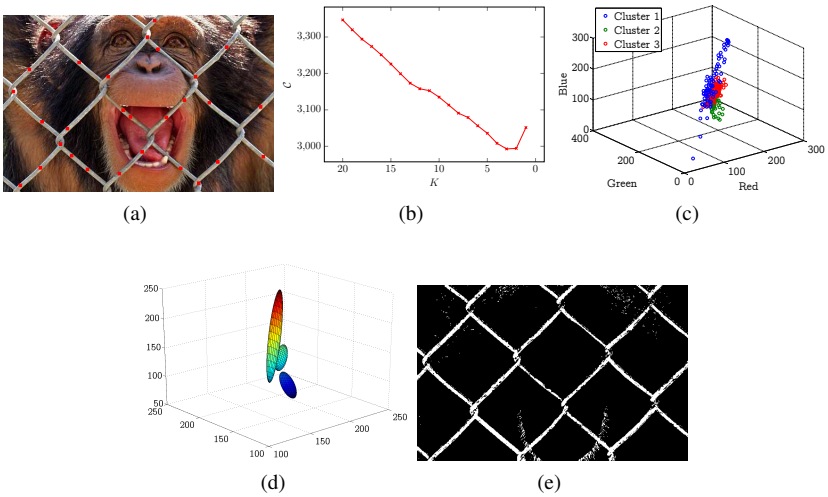
3.2 Initial Segmentation and Rectification

The Gaussian Mixture Model $\mathcal{G}$ is used to estimate the raw segmentation of the target object based on Mahalanobis distance [71], computed as:

$$d(\mathrm{x}, \mathcal{G}) = \sum_{i=1}^{K} \pi_i \sqrt{(\mathrm{x} - \mu_i)^\top \Sigma_i^{-1} (\mathrm{x} - \mu_i)} \tag{4}$$

For each image pixel x its distance $d(\mathrm{x}, \mathcal{G})$ from the estimated model is computed and it is classified as an object pixel if the distance is less than a pre-defined threshold τ, background pixel otherwise. Let Ω be the obtained object mask defined as:

$$\Omega(x,y) = \begin{cases} 1 & \text{if } d(I(x,y), \mathcal{G}) \leq \tau \\ 0 & \text{otherwise} \end{cases}$$



(a) (b) (c)

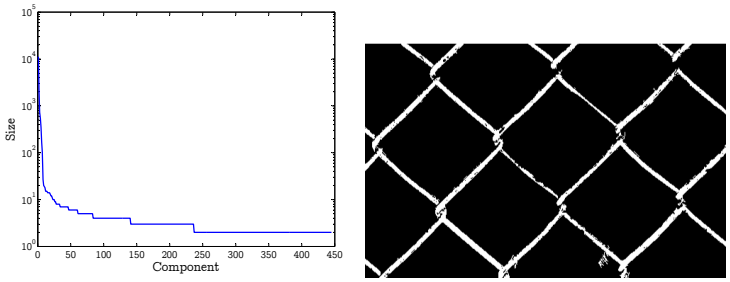
(d) (e)

Fig. 3 (a) Image with user input (red squares), (b) Model order estimation through DML method with EM algorithm, (c) Cluster wise data distribution, (d) The cluster ellipsoids showing the volume covered by each cluster in the RGB color space, (e) The initial segmentation map.

Here Ω represents the initial segmentation result. Fig. 3 shows an image with points selected by the user on the target object, i.e. the fence. From 28 input points the sample data is increased by including the corresponding 9-neighboring pixels resulting in 252 sample pixels. The GMM parameters and model order are iteratively estimated starting with $K_0 = 20$ clusters using EM and MDL approach as described in the previous section. Fig. 3b shows the Rissanen value $\mathcal{C}$ (Eq. 3) for different values of K: the minimum occurs at $K = 3$ which represents the optimal number of clusters for this example. Fig. 3c shows the cluster wise distribution of the sample points. Fig. 3d shows the corresponding clusters' ellipsoids and Fig. 3e shows the initial segmentation map Ω obtained with threshold $\tau = 1.2$.

It can be observed that the result of initial segmentation is not perfect; some non-target pixels are classified as object pixels (false positives) and object pixels are classified as background (false negatives). This can be seen in Fig. 3e where small islands of pixels appeared as a result of false positives. Further, a number of object regions, especially in the proximity of object boundaries, are classified as background. To remove the false positives, we use *Connected Component Analysis (CCA)* and to include the false negatives we utilize *Graph-Cuts*.

It is observed that false positives usually appear in very small size groups and true positives usually form large connected groups. To eliminate these small size groups we detect them by performing CCA [72] and computing the size of each component. The components are then sorted based on their size and top γ components are chosen while the rest are removed from the initial segmentation map. Fig. 4a shows the cardinality of each connected component found in Fig. 3e sorted in descending order. It can be seen that only first few components have large size and most components have size smaller than 10 pixels: these isolated islands are the false positives which



(a) (b)

Fig. 4 Connected component sizes sorted in descending order (a), Improved segmentation map after CCA with $\gamma = 6$ (b).

can be rejected safely. Fig. 4b shows the segmentation map after eliminating the false positives.

4 Segmentation Refinement

The initial segmentation suffers from false negatives which usually appear around the object boundaries and inside the segmentation map as well (see Fig. 4b). We use graph-cut techniques to tackle this issue and improve the segmentation results. Graph cut techniques have emerged as a powerful solutions to many optimization problems in computer vision including the object segmentation. In common graph-cut based methods the image pixels are divided into at least two and ideally three classes. We use the latter strategy and divide the image pixels into three classes namely, *definite foreground* F, *definite background* B and *unknown or uncertain* U. The first two classes F and B are considered as fixed whereas pixels belonging to U class are to be classified. This process is known as *Partitioning* and is based on the statistical characteristics of pixels in U class. Graph cut techniques solve this membership problem through Gaussian mixture models. GMMs for both fixed classes (F, B) are estimated and the problem of deciding the U class pixels is then solved though graph based optimization.

4.1 Graph Based Image Representation and Partitioning

The image is represented as an undirected weighted graph where the pixels form the vertices and edges link a vertex with its neighborhood. The weight of the edge represents the similarity between its end vertices. Two terminal nodes S and T are added in the graph representing the F and B classes respectively and they are linked to every other node in the graph. Now, there are two types of links that a vertex has: N-links and T-links. N-links are the links a vertex has with its neighborhood vertices whereas the T-links are the links of a vertex with the terminal nodes (S and T). The weight of N and T links computed as outlined in [38] and the fine segmentation of U is achieved by iterative energy minimization through max-flow min-cut algorithm [19].

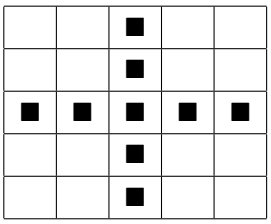
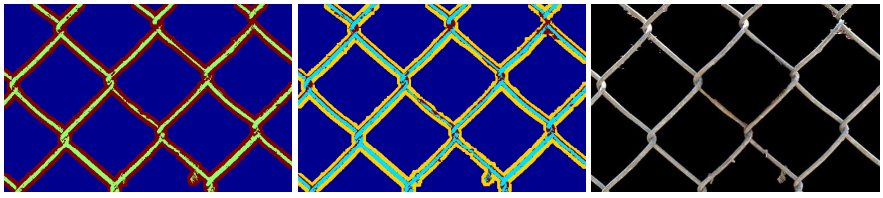


Fig. 5 A typical structuring element *cross* with $\omega = 5$.



(a) (b) (c)

Fig. 6 (a) Trimap: Green color represents the F region, Blue color shows the B region and dark red represents the U region. (b) The U region is partitioned into F (shown in dark red) and B (shown in yellow) using graph-cuts, (c) Final segmentation results.

To apply graph-cut approach to our problem, we need to define a trimap representing the corresponding classes: F, B and U. To this end, we take the segmentation map obtained from the previous section as F class. Since, the false negatives usually lie in and around the object boundary we use morphological operator *dilation* ($\oplus$) to estimate the 'Unknown' region U. The segmentation map Ω is dilated with structuring element s of size ω:

$$\Omega_{dil} = \Omega \oplus s \tag{5}$$

Fig. 5 shows a typical structuring element of size 5. The unknown region U region is then computed as:

$$U = \Omega_{dil} \setminus \Omega \tag{6}$$

and region B is computed by negating the dilated mask:

$$B = \texttt{NOT}(\Omega_{dil}) \tag{7}$$

A trimap $\mathcal{T}$ representing F pixels with 1, B pixels with 2 and U region with 0 is constructed and refined via graph cut method [19]. Fig. 6 shows the impact of dilation and final segmented image. Fig. 6a shows the trimap after dilation with structuring element of size 3×3. Fig. 6b shows the improvement achieved by graph-cut based refinement. Final segmented object is shown in Fig. 6c.

5 Experiments and Results

In this section, we evaluate the performance of the proposed algorithm on heterogeneous images with various types of objects which are subject to segmentation, e.g.,

Fig. 7 Image dataset used in experimental evaluation.

fences, railings, flowers, roads, and rivers. Qualitative and quantitative evaluation of the proposed method is also performed and the results are compared with the state-of-the-art object segmentation techniques.

5.1 Parameter Settings

Gaussian mixture model is considered to be the maturest statistical model, however its only shortcoming is its inability to find the optimal number of mixture components. We used MDL approach [64] in proposed algorithm to estimate the order, K, of GMM. From the experiments we found that the MDL estimated order of GMM generally solves the problem. Nonetheless, in few cases the number of mixture components was too large. Using too many components impacts on the overall running time of the algorithm and may also make it difficult to find good trade-offs in the selection of other parameters, e.g. the initial segmentation threshold τ. Nonetheless, this number can be used as an upper bound to the GMM order and user can select the best value by carefully analyzing the weights π of the mixture components. The components with very small weights, for example $\pi \leq 0.08$, do not add significant contribution to GMM and can be *merged* with other components. The parameter γ used in connected component analysis can be easily determined by examining the segmentation mask. The value of γ corresponds to the number of true components detected in the initial segmentation and can be chosen by inspecting the segmentation mask.

The size of the structuring element ω used in the dilation process for graph based segmentation refinement defines the disputed region, identified with U in Sec. 3, between the target object and the background regions. Our experiments show that the large value of ω results in large undefined region U which increases the processing time of the graph cut technique used in the refinement process. From experiments it is found that the best value of $\omega \in [5,20]$ with $\omega = 15$ being a good compromise in most cases.

Fig. 8 Segmentation results on images with fences. Top: input image, bottom: the segmentation achieved by the proposed algorithm.

5.2 Experimental Evaluation

To assess the performance of the proposed segmentation technique, a number of experiments are carried out on images of various kinds of distributed objects. The image dataset used in experimental evaluation is presented in Fig. 7. The dataset is designed aiming at testing the performance of the proposed algorithm on the images with challenging segmentation scenarios. It contains images with various types of distributed objects, where manual marking the target would be very difficult and time consuming, e.g., the images with regular and irregular pattern fences, and the aerial images of road and rivers. Our dataset also comprises images containing multiple distributed objects of the same kind, such as, flowers and sheep. More classical images with a single coherent object are included as a reference, as well. There are also aerial images of roads and rivers in the dataset to show the potential application of the proposed algorithm in remote sensing and geographical information systems.

The proposed DOST requires the user to specify a varying number of points of interest that depends on the color variation of the target object; for objects with limited color range 10 points are found to be sufficient whereas in case of large variation up to 20 clicks can be required. Anyhow, DOST is an interactive method where user can choose different number of points to improve the detection performance. In all experiments, the algorithm parameters are interactively tuned to optimize the segmentation results. In particular, the sample data is increased by including κ=5 neighboring pixels, connected component analysis is carried out with 8-neighbors. The clustering is performed using CLUSTER library [73] and K_0 in model order estimation is set to 20.

In the first set of experiments, fences from images are segmented as shown in Fig. 8. In each image, the fence is spread over the entire image and the image background contains various focused and de-focused regions. The proposed algorithm

Fig. 9 Segmentation results of various distributed objects. Top: input image, bottom: the segmentation achieved by the proposed algorithm.

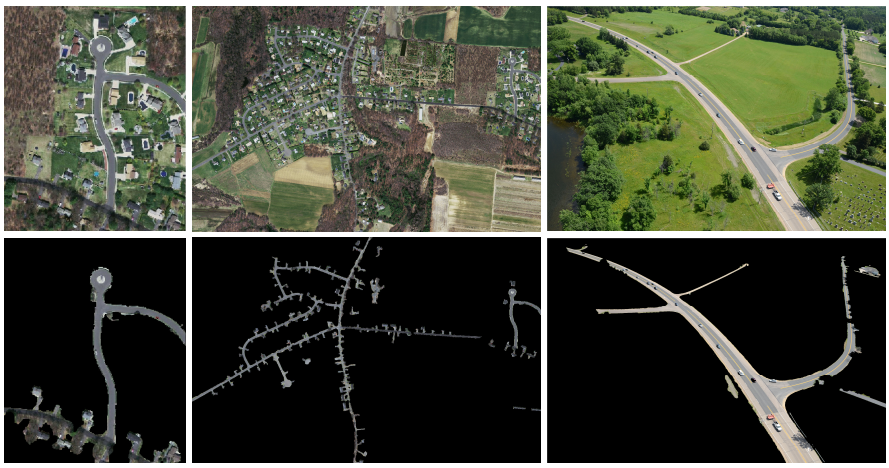


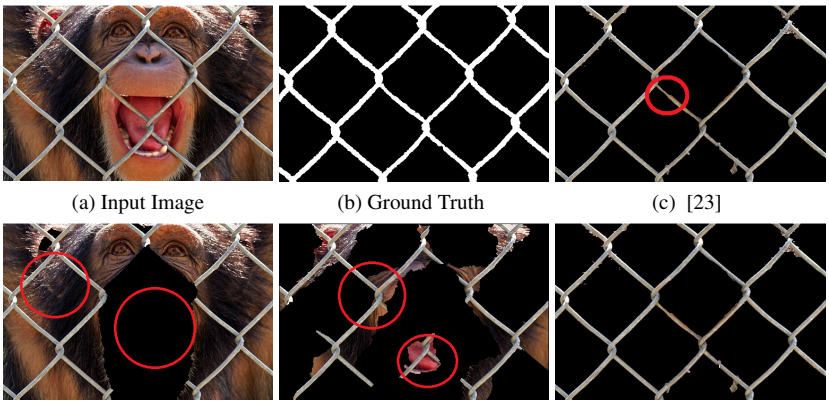
Fig. 10 Segmentation results of roads rivers from aerial images. The left two images are taken from the Massachusetts Roads Dataset [74]).

achieved promising results: in the two images on the left the fence is segmented without noticeable errors, whereas in the right image only few flower petals are erroneously selected due to their color similarity with the fence. In Fig. 9, the performance of the proposed technique is reported on distributed object segmentation. In the left image, irregular pattern fence is segmented. The image in the middle contains multiple distributed objects, flowers, and the target object in image on the right is a railing. The results presented in the bottom row of Fig. 9 show that the proposed algorithm is able to achieve a faithful segmentation.

Fig. 11 Segmentation results of rivers from aerial images.

Fig. 12 Segmentation of coherent objects.

In Fig. 10, we also show some remote sensing application examples where the proposed technique is used to segment roads, canals and rivers from aerial images. The images are captured at different heights allowing us to test the robustness of the proposed algorithm in challenging scenarios where road visibility is low. The two images on the left are taken from the Massachusetts Roads Dataset [74]. In Fig. 11, we test the algorithm on aerial images of rivers. In both experiments shown in Figs. 10 and 11, the proposed algorithm achieved convincing results. The performance of DOST on coherent objects is also tested and the results are presented in Fig. 12. The results show that the proposed technique is also effective for traditional object segmentation tasks. The results of the experiments presented above show the ability



(a) Input Image (b) Ground Truth (c) [23]

(d) [19] (e) [22] (f) Ours

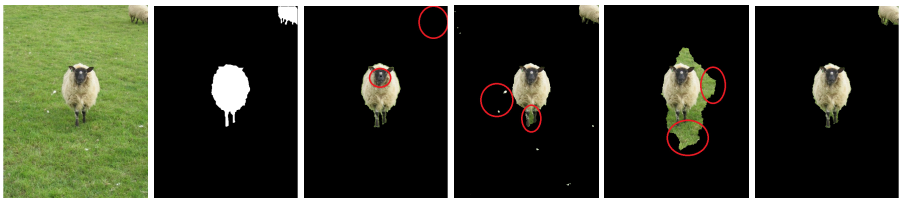
Fig. 13 Visual comparison of segmentation results of proposed tool and the compared methods on 'chimpanzee' test image. (a) Input image, (b) ground truth segmentation, (c) results of [23], (d) results of [19], (e) results of [22], and (f) our results.

of the proposed method to efficiently extract distributed objects from images with very little user assistance.

5.3 Qualitative Evaluation and Comparisons

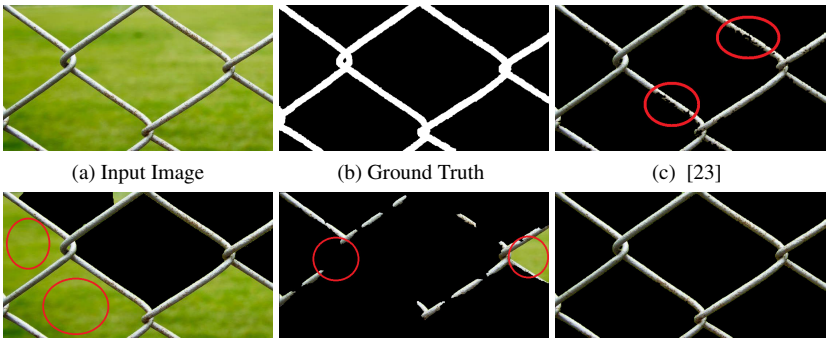
We also performed qualitative and quantitative evaluations of the proposed algorithm and compared the results with the well-known object segmentation approaches: GrabCut [19], GrowCut [22] and De-fencing method [23]. To perform quantitative evaluation, ground truth segmentation is required. To this end, we manually segment a subset of 8 images in our dataset, by carefully tracing the boundary of the target objects. In each experiment reported in the following, image (a) shows the input image and its ground truth is shown in (b). The results of [23], [19], and [22] are shown in (c), (d) and (e) respectively, and the results achieved by our method are shown in (f). The shortcomings or inaccuracies in the segmentation results are highlighted with red circles.

In the first experiment shown in Fig. 13, a visual comparison of the results achieved by the proposed algorithm and the compared methods is presented on the 'Chimpanzee' image from our dataset. The target object in this image is the fence which is thin and spread over the whole image. The segmentation achieved by algorithm [23] is close to the ground truth, however few parts of the fence, highlighted with a red circle, are not accurately segmented. The segmentation achieved by [19] is very poor since a large part of the image is detected along with the fence marks. The result of [22] is better than [19], but still regions around the fence are not correctly classified. Moreover, some segments of the fence are also missed. It can be noted that DOST segmentation does not produce significant artifacts, being very close to the ground truth.



(a) Input Image (b) Ground Truth (c) [23] (d) [19] (e) [22] (f) Ours

Fig. 14 Visual comparison of segmentation results of proposed tool and the compared methods on 'sheep' test image. (a) Input image, (b) ground truth segmentation, (c) results of [23], (d) results of [19], (e) results of [22], and (f) our results.



(a) Input Image (b) Ground Truth (c) [23]

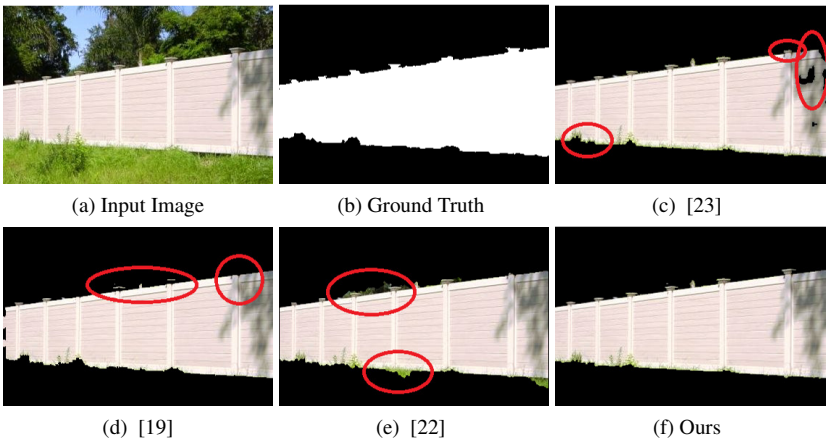
(d) [19] (e) [22] (f) Ours

Fig. 15 Visual comparison of segmentation results of proposed tool and the compared methods on 'park' test image. (a) Input image, (b) ground truth segmentation, (c) results of [23], (d) results of [19], (e) results of [22], and (f) our results.

In Fig. 14, we perform visual comparisons on test image 'Sheep' containing two target objects. The segmentation results achieved by [23] and [22] missed the smaller sheep in the image; in the results of [23], there is a small hole in the face of the detected sheep; the segmentation achieved by [22] is not accurate, particularly around the contours of the target object. The results of [19] and our method are quite good: both sheep are detected and the segmentation is precise. However, in the results of [19], few small blobs from background are categorized as object; our results are almost error-free.

In the next experiment, Fig. 15, a fence object is subject to segmentation. The segmentation results of [19] are very poor: the background in three fence squares could not be separated from fence; the results achieved by [22] are better than [19]: however some fence segments are not detected and a small background region is incorrectly detected. The results of [23] are close to ground truth except few parts of the fence highlighted with red circles. The segmentation achieved by our method are very accurate also in this case.

Figure 16 presents the visual comparison of the results achieved by our technique and the compared methods on the 'Wall' test image. The segmentation achieved by



(a) Input Image (b) Ground Truth (c) [23]

(d) [19] (e) [22] (f) Ours

Fig. 16 Visual comparison of segmentation results of proposed tool and the compared methods on 'wall' test image. (a) Input image, (b) ground truth segmentation, (c) results of [23], (d) results of [19], (e) results of [22], and (f) our results.

[23] suffers from holes in the wall, marked with red circle, appeared due to incorrect classification of the target object. The results achieved by [19] and [22] are better than [23]: however there are few inaccuracies in their results around the contours of the wall. The segmentation achieved by our method is very precise and does not exhibits such flaws.

Further qualitative evaluation and comparisons can be found on the paper web site[1], which are not presented here to limit the length of the paper. The qualitative evaluation show that the results of the algorithm proposed in [23] are generally good for all test images. The results of [19] and [22] are good for images with coherent objects, however their performance turns to be rather poor in presence of distributed objects like thin fences. We noted that such impairment is mainly determined by the method used to collect the user input: drawing a rectangle around the target object is not always feasible and also specifying the input by marking few lines on the target object and on the background may be cumbersome in some distributed object cases. The performance of the algorithm proposed in [23] is better than other compared methods, however the results presented above show that the segmentation achieved by [23] may be imprecise, particularly around the boundaries of the foreground objects. On the other hand the results of the algorithm proposed in this work are always quite accurate.

5.4 Objective Evaluation and Comparison

In this section, we evaluate the performance of the proposed algorithm quantitatively and also compare the results with the techniques proposed in [19], [22], and [23].

[1] `http://www.di.unito.it/~farid/Research/DOST.html`

5.4.1 Objective Evaluation Parameters

A ground truth indeed classifies the image pixels into two classes - $P(Positives)$ and $N(Negatives)$. The P class represents the target object pixels and N contains the rest. The performance of a binary segmentation algorithm can be analyzed in terms of *accuracy* and *precision* [75–77]. There are four possible outcomes when a binary segmentation $S' = \{P', N'\}$ obtained through a classifier is compared to the corresponding ground truth image $S = \{P, N\}$:

1. *True Positives (TP)*: the object pixels which are correctly classified as object i.e. $TP = P' \cap P$.
2. *False Negatives (FN)*: the object pixels which are incorrectly marked as non-object i.e. $FN = P' - P$.
3. *True Negatives (TN)*: the non-object pixels which are correctly classified as non-object. $TN = N' \cap N$.
4. *False Positives (FP)*: the non-object pixels that are incorrectly labeled as object. $FP = N' - N$.

Furthermore, the previous scores can be arranged in the so called *confusion matrix* that can be used to rank the overall performance. We compute various statistical measures to evaluate the performance of the proposed technique. *Sensitivity* and *Specificity* are two basic performance measures that compute the rate of correctly classified positives and correctly classified negatives respectively. Sensitivity (also known as Recall and True Positive Rate (TPR)) measures the proportion of correctly identified positives to actual positives. The Specificity (also called True Negative Rate (TNR)) is the ratio of correctly classified negatives to actual negatives. These measures are computed as follows:

$$Recall = \frac{|TP|}{|P|}; \quad Specificity = \frac{|TN|}{|N|} \tag{8}$$

Large values of *recall* and $specificity$ indicate better classification. We also compute *Accuracy* and *Precision* metrics that characterise the degree of closeness and repeatability respectively:

$$Accuracy = \frac{|TP| + |TN|}{|P| + |N|}; \quad Precision = \frac{|TP|}{|TP| + |FP|} \tag{9}$$

However, *accuracy* and *precision* metrics can be delusive in cases when the positive P and negative N classes are highly unbalanced. To this end, $F_1 score$ is computed which is considered to be more reliable. It measures the Harmonic mean of *recall* and *precision* and is computed as follows:

$$F_1 score = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} \tag{10}$$

Table 1 Objective performance analysis of the proposed and the compared methods in terms of Accuracy and Precision. The best results are in bold.

Exp.	Accuracy				Precision			
	[23]	[19]	[22]	Ours	[23]	[19]	[22]	Ours
Chimpanzee	0.96	0.32	0.80	**0.98**	**0.96**	0.14	0.37	0.91
Sparrow	0.95	**0.97**	0.71	0.96	**0.98**	0.50	0.42	0.90
Sheep	0.93	**0.99**	0.92	0.98	0.85	**0.94**	0.45	0.88
Snapdragon	0.97	0.86	0.84	**0.99**	0.88	0.39	0.32	**0.97**
Park	0.95	0.57	0.88	**0.99**	0.98	0.24	0.63	**0.99**
Rose	0.95	**1.00**	0.99	0.99	0.85	**1.00**	0.99	0.92
Wall	0.96	**0.98**	**0.98**	**0.98**	0.85	**0.97**	0.96	**0.97**
Bear	0.96	**0.99**	**0.99**	**0.99**	0.95	0.95	**0.99**	0.97
Avg.	0.95	0.84	0.89	**0.98**	0.91	0.64	0.64	**0.94**

5.4.2 Objective Evaluation Results

The segmentation accuracy metrics presented in the previous section have been computed for all experiments described in this paper. The results are reported in Tables 1 and 2, with the best performance formatted in bold. The Accuracy and Precision statistics for each image are presented in Tab. 1. In terms of accuracy, on three test images, Sparrow, Sheep and Rose, the algorithm [19] performs marginally better than our method. On the rest of the images, our algorithm outperforms all the compared methods. The average segmentation accuracy of [19], [22], and [23] is 0.84, 0.89, and 0.95, respectively. The proposed method achieves the best average accuracy of 0.98 outperforming the compared methods up to a margin of 15%. The results in terms of precision parameter are somewhat mixed: on two images algorithm [23] performs better than others, on three images [19] performs the best. Algorithm [22] achieves the best precision on one image and our method outperforms the compared methods on three test images. The average precision of [19], [22], and [23] is 0.64, 0.64, and 0.91, respectively. The average precision achieved by our method is 0.94, significantly better than the compared methods.

The Recall and F1 Score analysis is presented in Tab. 2. The statistics show that on two images algorithm [19] achieves the best recall and on the rest of the tests our method beats the others. There is one image where [22], [19] and the proposed method achieves the same recall measure. However, in terms of average recall, the proposed algorithm performs the best by attaining a score of 0.97. The closest is [19] with 0.95. The average recall of [23] and [22] are low, 0.92 and 0.80, respectively. The F1 score of [19] is the best on three images, on the other tests the proposed approach outperforms all the competing algorithms. Algorithm [22] performs the poorest in terms of average F1 score, 0.69. The average F1 score of [23] and [19] is 0.91 and 0.72, respectively. The average F1 score of the proposed method turns to be the highest among the competing method with a value of 0.95.

Table 2 Objective performance analysis of the proposed and the compared methods in terms of Recall and F1 Score. The best results are in bold.

Exp.	Recall				F1 Score			
	[23]	[19]	[22]	Ours	[23]	[19]	[22]	Ours
Chimpanzee	0.88	0.75	0.69	**0.96**	0.92	0.24	0.49	**0.93**
Sparrow	0.76	**0.93**	**0.93**	**0.93**	0.86	0.65	0.58	**0.91**
Sheep	0.88	**0.98**	0.83	0.91	0.86	**0.96**	0.59	0.89
Snapdragon	0.96	0.95	0.67	**0.96**	0.92	0.55	0.43	**0.96**
Park	0.97	0.98	0.36	**1.00**	0.97	0.39	0.46	**0.99**
Rose	0.96	0.99	0.96	**1.00**	0.90	**0.99**	0.98	0.96
Wall	0.96	0.98	0.99	**1.00**	0.90	**0.99**	0.98	0.98
Bear	0.96	**1.00**	0.95	0.99	0.95	0.97	0.97	**0.98**
Avg.	0.92	0.95	0.80	**0.97**	0.91	0.72	0.69	**0.95**

Both the quantitative and the qualitative results show the effectiveness of the proposed tool in efficiently segmenting the distributed and scattered objects as well as the coherent objects from images. Application of the proposed tool in remote sensing to segment roads and rivers in aerial images is also demonstrated.

6 Conclusions

In this paper we proposed a tool for segmentation of objects scattered over the whole image. Due to peculiar shapes e.g. fences, railings, rivers, roads, segmentation of such objects is a challenging task under many aspects: easy and minimum user interaction in marking the seeds for the target object in addition to accurate and precise segmentation. The proposed solution exploits statistical characteristics of the target object to learn mixture of Gaussians which are then used for segmentation. To ensure minimum user intervention the input method is simplified in the form of few mouse clicks. Moreover, the number of components in the mixture are estimated through MDL method. The initial segmentation may be polluted with false positives which are removed through connected component analysis. Finally, accurate segmentation is guaranteed through graph cuts used to recover the missed regions. The obtained subjective and objective results show the effectiveness of the proposed segmentation framework.

References

1. Dzung L Pham, Chenyang Xu, and Jerry L Prince. Current methods in medical image segmentation. *Annu. Rev. Biomed. Eng.*, 2(1):315–337, 2000.
2. R. Wang, J. Lv, and S. Ma. A mri image segmentation method based on medical semaphore calculating in medical multimedia big data environment. *Multimedia Tools and Applications*, pages 1–21, 2017.
3. W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *ACM Comput. Surv.*, 35(4):399–458, December 2003.

4. S. Belongie, C. Carson, H. Greenspan, and J. Malik. Color- and texture-based image segmentation using em and its application to content-based image retrieval. In *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, pages 675–682, Jan 1998.
5. G. Wang, J. Lu, Z. Pan, and Q. Miao. Color texture segmentation based on active contour model with multichannel nonlocal and tikhonov regularization. *Multimedia Tools and Applications*, pages 1–12, 2016.
6. Y. Wu, X. Peng, K. Ruan, and Z. Hu. Improved image segmentation method based on morphological reconstruction. *Multimedia Tools and Applications*, pages 1–13, 2016.
7. C. Yan, H. Xie, D. Yang, J. Yin, Y. Zhang, and Q. Dai. Supervised hash coding with deep neural network for environment perception of intelligent vehicles. *IEEE Trans. Intell. Transp. Syst.*, PP(99):1–12, 2017.
8. M. H. Khan, K. Shirahama, M. S. Farid, and M. Grzegorzek. Multiple human detection in depth images. In *Proc. Int. Workshop Multimed. Signal Process. (MMSP)*, pages 1–6, Sept 2016.
9. H. Tazeem, M.S. Farid, and A. Mahmood. Improving security surveillance by hidden cameras. *Multimedia Tools and Applications*, 76(2):2713–2732, 2017.
10. J Im, JR Jensen, and JA Tullis. Object-based change detection using correlation image analysis and image segmentation. *Int. J. Remote Sens.*, 29(2):399–423, 2008.
11. C. Yan, H. Xie, S. Liu, J. Yin, Y. Zhang, and Q. Dai. Effective uyghur language text detection in complex background images for traffic prompt identification. *IEEE Trans. Intell. Transp. Syst.*, PP(99):1–10, 2017.
12. FranciscoJ. Hernandez-Lopez and Mariano Rivera. Change detection by probabilistic segmentation from monocular view. *Mach. Vis. Appl.*, 25(5):1175–1195, 2014.
13. Marco Cagnazzo, Sara Parrilli, Giovanni Poggi, and Luisa Verdoliva. Costs and advantages of object-based image coding with shape-adaptive wavelet transform. *EURASIP J. Image Video Process.*, 2007(1):19–19, January 2007.
14. H. Shen, W. D. Pan, and D. Wu. Predictive lossless compression of regions of interest in hyperspectral images with no-data regions. *IEEE Trans. Geosci. Remote Sens.*, 55(1):173–182, Jan 2017.
15. C. Yan, Y. Zhang, J. Xu, F. Dai, J. Zhang, Q. Dai, and F. Wu. Efficient parallel framework for HEVC motion estimation on many-core processors. *IEEE Trans. Circuits Syst. Video Technol.*, 24(12):2077–2089, Dec 2014.
16. C. Yan, Y. Zhang, J. Xu, F. Dai, L. Li, Q. Dai, and F. Wu. A highly parallel framework for HEVC coding unit partitioning tree decision on many-core processors. *IEEE Signal Process. Lett.*, 21(5):573–576, May 2014.
17. X. Xu, W. Geng, R. Ju, Y. Yang, T. Ren, and G. Wu. OBSIR: Object-based stereo image retrieval. In *Proc. IEEE Int. Conf. Multimed. and Expo (ICME)*, pages 1–6, July 2014.
18. Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David Shamma, Michael Bernstein, and Li Fei-Fei. Image retrieval using scene graphs. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, June 2015.
19. Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. "GrabCut": Interactive Foreground Extraction Using Iterated Graph Cuts. *ACM Trans. Graph.*, 23(3):309–314, August 2004.
20. Qiong Yang, Xiaoou Tang, Chao Wang, Zhongfu Ye, and Mo Chen. Progressive Cut: An Image Cutout Algorithm that Models User Intentions. *IEEE Trans. Multimedia*, 14(3):56–66, July 2007.
21. Y.Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary amp; region segmentation of objects in n-d images. In *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, volume 1, pages 105–112 vol.1, 2001.
22. Vladimir Vezhnevets and Vadim Konouchine. Growcut: Interactive multi-label nd image segmentation by cellular automata. In *Proc. of Graphicon*, pages 150–156, 2005.
23. M.S. Farid, A. Mahmood, and M. Grangetto. Image de-fencing framework with hybrid inpainting algorithm. *Signal Image Video Process.*, 10(7):1193–1201, 2016.
24. Yung-Yu Chuang, B. Curless, D.H. Salesin, and R. Szeliski. A bayesian approach to digital matting. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, volume 2, pages 264–271, 2001.
25. Eric N. Mortensen and William A. Barrett. Interactive segmentation with intelligent scissors. *Graph Model Im. Proc.*, 60(5):349 – 384, 1998.
26. Jue Wang and Michael F Cohen. *Image and video matting: a survey*. Now Publishers Inc, 2008.
27. M.S. Farid, M. Lucenteforte, M.H. Khan and M. Grangetto. Semi-automatic Segmentation of Scattered and Distributed Objects. In *Proc. 10th Int. Conf. on Computer Recognition Systems (CORES)*, pages 110–119, 2017.
28. Stanley Osher and James A Sethian. Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations. *J. Comput. Phys.*, 79(1):12 – 49, 1988.

29. G. Kuntimad and H.S. Ranganath. Perfect image segmentation using pulse coupled neural networks. *IEEE Trans. Neural Netw.*, 10(3):591–598, May 1999.
30. S. Beucher and F. Meyer. The morphological approach to segmentation: the watershed transformation. Mathematical morphology in image processing. *Opt. Eng.*, 34:433–481, 1993.
31. Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(11):1222–1239, Nov 2001.
32. M.A. Ruzon and C. Tomasi. Alpha estimation in natural images. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, volume 1, pages 18–25, 2000.
33. Eric N. Mortensen and William A. Barrett. Intelligent scissors for image composition. In *Proc. 22nd Annu. Conf. Comput. Graph. Interact. Tech.*, SIGGRAPH '95, pages 191–198. ACM, 1995.
34. E. Mortensen, B. Morse, W. Barrett, and J. Udupa. Adaptive boundary detection using 'live-wire' two-dimensional dynamic programming. In *Proc. Comput. Cardiol.*, pages 635–638, Oct 1992.
35. A. Berman, A. Dadourian, and P. Vlahos. Method for removing from an image the background surrounding a selected object, October 17 2000. US Patent 6,134,346.
36. A. Berman, P. Vlahos, and A. Dadourian. Method for removing from an image the background surrounding a selected subject by generating candidate mattes, September 11 2001. US Patent 6,288,703.
37. M.F. Tappen and W.T. Freeman. Comparison of graph cuts with belief propagation for stereo, using identical MRF parameters. In *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, pages 900–906, Oct 2003.
38. Yuri Boykov and Gareth Funka-Lea. Graph Cuts and Efficient N-D Image Segmentation. *Int. J. Comput. Vis.*, 70(2):109–131, 2006.
39. Bo Peng, Lei Zhang, and David Zhang. A survey of graph theoretical approaches to image segmentation. *Pattern Recognit.*, 46(3):1020 – 1038, 2013.
40. Yin Li, Jian Sun, Chi-Keung Tang, and Heung-Yeung Shum. Lazy snapping. *ACM Trans. Graph.*, 23(3):303–308, August 2004.
41. E.N. Mortensen and W.A. Barrett. Toboggan-based intelligent scissors with a four-parameter edge model. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, volume 2, pages 452–458, 1999.
42. Jue Wang and M.F. Cohen. An iterative optimization approach for unified image segmentation and matting. In *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, volume 2, pages 936–943, Oct 2005.
43. J. H. Bosamiya, P. Agrawal, P. P. Roy, and R. Balasubramanian. Script independent scene text segmentation using fast stroke width transform and grabcut. In *3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 151–155, Nov 2015.
44. Eyasu Zemene and Marcello Pelillo. Interactive image segmentation using constrained dominant sets. In *Proc. 14th European Conference Computer Vision (ECCV)*, pages 278–294, Cham, 2016. Springer International Publishing.
45. M. M. Cheng, V. A. Prisacariu, S. Zheng, P. H. S. Torr, and C. Rother. Densecut: Densely connected crfs for realtime grabcut. *Computer Graphics Forum*, 34(7):193–201, 2015.
46. Disi Chen, Gongfa Li, Ying Sun, Jianyi Kong, Guozhang Jiang, Heng Tang, Zhaojie Ju, Hui Yu, and Honghai Liu. An interactive image segmentation method in hand gesture recognition. *Sensors*, 17(2), 2017.
47. John von Neumann. The general and logical theory of automata. *Cerebral Mechanisms in Behavior – The Hixon Symposium*, pages 1–31, 1951.
48. O. Juan and Y. Boykov. Active Graph Cuts. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, volume 1, pages 1023–1029, June 2006.
49. V. Vineet and P. J. Narayanan. CUDA cuts: Fast graph cuts on the GPU. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshop (CVPRW)*, pages 1–8, June 2008.
50. P. Strandmark and F. Kahl. Parallel and distributed graph cuts by dual decomposition. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 2085–2092, June 2010.
51. A. Delong and Y. Boykov. A scalable graph-cut algorithm for N-D grids. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 1–8, June 2008.
52. Yi Peng, Li Chen, Fang-Xin Ou-Yang, Wei Chen, and Jun-Hai Yong. JF-Cut: A parallel graph cut approach for large-scale image and video. *IEEE Trans. Image Process.*, 24(2):655–666, Feb 2015.
53. Jiangyu Liu and Jian Sun. Parallel graph-cuts by adaptive bottom-up merging. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 2181–2188, June 2010.
54. V. Kolmogorov and R. Zabin. What energy functions can be minimized via graph cuts? *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(2):147–159, Feb 2004.
55. Y. Boykov and O. Veksler. Graph cuts in vision and graphics: Theories and applications. In Nikos Paragios, Yunmei Chen, and Olivier Faugeras, editors, *Handbook of Mathematical Models in Computer Vision*, pages 79–96. Springer US, 2006.

56. D. Ren, Z. Jia, J. Yang, and N. K. Kasabov. A practical grabcut color image segmentation based on bayes classification and simple linear iterative clustering. *IEEE Access*, 5:18480–18487, 2017.
57. R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Ssstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(11):2274–2282, Nov 2012.
58. D. Lin, J. Dai, J. Jia, K. He, and J. Sun. ScribbleSup: Scribble-supervised convolutional networks for semantic segmentation. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 3159–3167, June 2016.
59. Q. Guan, M. Hua, and H. G. Hu. A modified grabcut approach for image segmentation based on local prior distribution. In *2017 International Conference on Wavelet Analysis and Pattern Recognition (ICWAPR)*, pages 122–126, July 2017.
60. A. Heimowitz and Y. Keller. Image segmentation via probabilistic graph matching. *IEEE Trans. Image Process.*, 25(10):4743–4752, Oct 2016.
61. M. Jian and C. Jung. Interactive image segmentation using adaptive constraint propagation. *IEEE Trans. Image Process.*, 25(3):1301–1311, March 2016.
62. Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *J. R. Stat. Soc. Ser. B-Stat. Methodol.*, pages 1–38, 1977.
63. R. J. A. Little and D. B. Rubin. *Statistical Analysis with Missing Data*. Wiley Series in Probability and Statistics. Wiley, New York, 1st edition, 1987.
64. Jorma Rissanen. A universal prior for integers and estimation by minimum description length. *Ann. Stat.*, 11(2):416–431, 06 1983.
65. Richard A Redner and Homer F Walker. Mixture densities, maximum likelihood and the em algorithm. *SIAM Rev.*, 26(2):195–239, 1984.
66. R.S. Wallace and T. Kanade. Finding natural clusters having minimum description length. In *Proc. Int. Conf. Pattern Recognit. (ICPR)*, volume i, pages 438–442, Jun 1990.
67. Mark H Hansen and Bin Yu. Model selection and the principle of minimum description length. *J. Am. Stat. Assoc.*, 96(454):746–774, 2001.
68. Petri Kontkanen, Petri Myllymki, Wray Buntine, Jorma Rissanen, and Henry Tirrii. An MDL Framework for Data Clustering, 2003.
69. Peter Grnwald. A Tutorial Introduction to the Minimum Description Length Principle, 2005.
70. J. Rissanen. Strong optimality of the normalized ml models as universal codes and information in data. *IEEE Trans. Inf. Theory*, 47(5):1712–1717, Jul 2001.
71. P. C. Mahalanobis. On the generalised distance in statistics. *Proceedings National Institute of Science, India*, 2(1):49–55, April 1936.
72. George Stockman and Linda G. Shapiro. *Computer Vision*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition, 2001.
73. C. A. Bouman. Cluster: An unsupervised algorithm for modeling Gaussian mixtures. Available from http://engineering.purdue.edu/˜bouman, April 1997.
74. V. Mnih. Massachusetts roads dataset. Available from http://www.cs.toronto.edu/ vmnih/data/, 2013.
75. Jayaram K Udupa, Vicki R LaBlanc, Hilary Schmidt, Celina Imielinska, Punam K Saha, George J Grevera, Ying Zhuge, LM Currie, Pat Molholt, and Yinpeng Jin. Methodology for evaluating image-segmentation algorithms. In *Medical Imaging*, pages 266–277, 2002.
76. Jayaram K. Udupa, Vicki R. LeBlanc, Ying Zhuge, Celina Imielinska, Hilary Schmidt, Leanne M. Currie, Bruce E. Hirsch, and James Woodburn. A framework for evaluating image segmentation algorithms. *Comput. Med. Imaging Graph.*, 30(2):75 – 87, 2006.
77. Tom Fawcett. An Introduction to ROC Analysis. *Pattern Recognit. Lett.*, 27(8):861–874, June 2006.